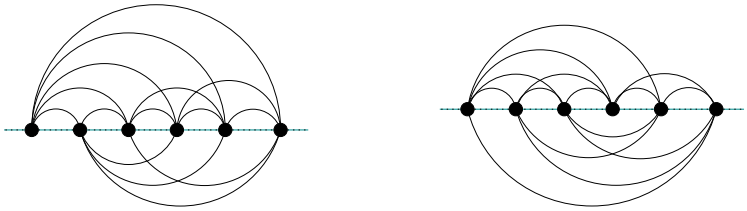
A Simulated Annealing Algorithm for the 2-page Crossing Number Problem

Timo Poranen, *Dept. of CS, P.O. Box 607, FIN-33014, Univ. Tampere, Finland*
Erkki Mäkinen, *Dept. of CS, P.O. Box 607, FIN-33014, Univ. Tampere, Finland*
Hongmei He, *Dept. of CS, Loughborough Univ., Loughborough, Leicestershire LE11 3TU, UK*

Keywords: book drawings, (fixed) linear crossing number, 2-page crossing number, simulated annealing

1. Introduction

The simplest graph drawing method is that of putting the vertices of a graph on a line (spine) and drawing the edges as half-circles in κ half planes. Such drawings are called κ-page book drawings. Edge crossing minimisation is the most important goal in linear Very Large Scale Integration (VLSI) circuit design and Quantum-dot Cellular Automata (QCA), since a smaller number of crossings means lower cost. The minimum number of edge crossings over all κ-page book drawings of a graph is called the κ-page book crossing number of the graph [17].



(a) Five crossings (b) Three crossings

Figure 1: 2-page drawings of K_6 with five and three crossings.

In a 1-page book drawing, all edges are placed on one side of the spine, and in a 2-page book drawing, all edges are placed either upon or below the spine. The minimum number of edge crossings over all 1-page (resp. 2-page) drawings of a graph G is called the 1-page (resp. 2-page) crossing number of G, denoted as $\nu_1(G)$ (resp. $\nu_2(G)$). The 1-page crossing number problem is to find a good order of vertices, and the 2-page crossing number problem is further to find a good edge distribution in the two pages. Both problems have been proved to be NP-hard [11, 12]. Figure 1 illustrates 2-page drawings of K_6, for which $\nu_2(G) = 3$, with five (Fig. 1 (a)) and three crossings (Fig. 1 (b)).

A lot of heuristic algorithms were designed for the 1-page and 2-page crossing number problem, see for instance [1, 6, 15, 16, 18] for the 1-page and [2, 3, 4, 8, 9, 13, 16] for the 2-page crossing number problem.

In this paper we design a simulated annealing algorithm starting with a good seed by the algorithm of Baur and Brandes [1] with the edge length strategy of dividing edges into two sets [3, 9]. The experimental data show that the simulated annealing algorithm achieves comparable or better results than evolution algorithms presented in the literature [7, 8].

2. Previous algorithms

The best algorithm by Baur and Brandes [1] for the 1-page crossing number problem consists of greedy and sifting phases. At each step of the greedy phase a vertex with the largest number of already placed neighbours

is placed to the end that yields fewer crossings of edges being closed with open edges (where an open edge refers to an edge with an end not yet selected).

In the sifting phase, each vertex is moved along a fixed order of all other vertices. The vertex is then placed in its locally optimal position. The greedy and sifting phases can be implemented to run in $O(nm)$ time for a graph with n vertices and m edges.

Cimikowski [3] proposed an edge length heuristic for the fixed linear crossing number problem. Suppose that an initial permutation of the vertices is given. The heuristic first orders all edges non-increasingly by their length in a list. Then edges are placed in the list order to a page where fewer crossings will be produced.

By applying first the algorithm by Baur and Brandes to construct an initial permutation and then the edge length strategy to divide edges into two pages, gave good approximations in the experiments by He et al. [9]. We denote this algorithm by BB+len.

Algorithm 1 A simulated annealing algorithm for the 2-page crossing number problem.

Select a cooling ratio α and an initial temperature t_0. Select a frozen temperature t_1 and an equilibrium detection rate r. Set a counter $c \leftarrow 0$ and $t \leftarrow t_0$;
Construct an initial solution $s_0 = (\pi, P_1, P_2)$ for the input graph G. That is, find an initial permutation π for the vertices on the spine and divide the edges into the two pages P_1 and P_2;
while $t \geq t_0$ and $crossings(\pi, P_1, P_2) > 0$ **do**
  while $c \leq 2r$ and $crossings(\pi, P_1, P_2) > 0$ **do**
    $c \leftarrow c + 1$;
    if $c \leq r$ **then**
      Randomly select a side $s \in \{1, 2\}$ and an edge e from P_s;
      Set $\delta \leftarrow crossings(\pi, P_s \setminus \{e\}, P_{k \neq s, k \in \{1,2\}} \cup \{e\}) - crossings(\pi, P_1, P_2)$;
      if $\delta \leq 0$ **then**
        Set $s_0 \leftarrow (\pi, P_s \setminus \{e\}, P_{k \neq s, k \in \{1,2\}} \cup \{e\})$;
      else
        Generate a random real i, $0 \leq i \leq 1$;
        if $i \leq e^{-\delta/t}$ **then**
          Set $s_0 \leftarrow (\pi, P_s \setminus \{e\}, P_{k \neq s, k \in \{1,2\}} \cup \{e\})$;
        end if
      end if
    else {Now we have $r < c \leq 2r$}
      Randomly select two vertices on the spine and swap their positions to obtain permutation π';
      Set $\delta \leftarrow crossings(\pi', P_1, P_2) - crossings(\pi, P_1, P_2)$;
      if $\delta \leq 0$ **then**
        Set $s_0 \leftarrow (\pi', P_1, P_2)$;
      else
        Generate a random real i, $0 \leq i \leq 1$;
        if $i \leq e^{-\delta/t}$ **then**
          Set $s_0 \leftarrow (\pi', P_1, P_2)$;
        end if
      end if
    end if
  end while
  $t \leftarrow \alpha t$;
  $c \leftarrow 0$;
end while
return The best found solution over the whole optimisation process;

3. The simulated annealing algorithm

Simulated annealing (SA) optimisation method imitates the cooling process of material in a heat bath. The main principles of the SA were first introduced in an article by Metropolis et al. [14] and later generalised by Kirkpatrick, Gelatt and Vecchi [10]. The basic idea behind the SA algorithms is to choose an *initial solution*, denoted by s_0, and then to try to improve this solution.

Our implementation of the SA algorithm (see Algorithm 1) for minimising the number of crossings in 2-page drawings is based on maintaining two sets of edges (pages P_1 and P_2 of the drawing) and a permutation π of vertices on the spine. In what follows, the current solution is denoted by $s_0 = (\pi, P_1, P_2)$ and the number of crossings in the solution s_0 is denoted by $crossings(s_0)$. In Steps $6-16$ the algorithm randomly changes the pages of the edges of the current solution, and the change is accepted if it improves the current solution or the condition in Step 13 is fullfilled: Let s' be a new solution and let $\delta = crossings(s') - crossings(s_0)$. If $\delta > 0$ (s' is worse than s_0), then s' is accepted to be the new current solution if $i \leq e^{-\delta/t}$, where i is a random real, $0 \leq i \leq 1$, and t is the current temperature. Between Steps $17-18$ the algorithm randomly changes the positions of the vertices on the spine. The changes are accepted if they improve the current solution or the condition (similar to the one in Step 13) in Step 24 is satisfied. When the frozen temperature is reached, the algorithm returns the best found solution over the whole optimisation process.

Whether we try to move edges in the sets or vertices on the spine, depends on how close we are equilibrium state. If counter is $\leq r$, we try to move edges, and when counter $> r$ and $\leq 2r$, we try to move vertices.

4. Experiments

We implemented the simulated annealing algorithm in C++. The test runs were executed on a one-processor AMD Athlon $1GHz$ with 256 Megabytes memory computer (1992 BogoMips) running under Linux Mandrake 8.1.

We test the simulated annealing algorithm for 20 random biconnected planar Rome graphs [5] with 50 vertices and for 34 circulant graphs [3, 7]. Circulant graphs of the form $C_n(a_1, a_2, ..., a_k)$, where $0 < a_1 < a_2 < ... < a_k < (n+1)/2$, are regular Hamiltonian graphs with n vertices, and with vertices $i \pm a_j (\text{mod } n)$, $j = 1..k$, adjacent to each i.

Since the best known results for the 2-page crossing number problem have been produced by genetic algorithms [7], we compared the SA solutions against them.

Table 1 shows the solutions for the circulant graphs, obtained by two different genetic algorithms by He et al. [7] (GA_2P and GA_2Ptg), generated randomly (RND), constructed by BB+len and by our simulated annealing algorithm solutions (SA) with different initial parameters (RND or BB+len) and cooling ratios ($\alpha = 0.9$, $\alpha = 0.995$ and $\alpha = 0.999$). Other parameters were $t_0 = 1.0$, $t_1 = 0.2$ and $r = n^2$, where n is the number of vertices in the graph. Table 2 shows similar solutions for the Rome graphs.

The chosen parameters yield running time $O(n^2m^2)$ for a graph with n vertices and m edges, since the inner while loop is executed $2n^2$ times, and our naive calculation for the number of crossings takes $O(m^2)$ time. The constant factor depends on the chosen annealing parameters. Our implementation for the SA algorithm is not optimal, and its running time can be further improved by using an intersection graph data structure [19].

The random solutions are clearly the worst. The quality of solutions of the SA depends highly on the initial solution and the cooling parameters. When a random initial solution is used as a starting solution, the final approximations are on average worse than solutions when using BB+len solutions as initial solutions, this can be seen from columns seven and eight of Table 1.

The larger the parameter α is, the longer time the cooling process takes, but the algorithm produces better approximations. With $\alpha = 0.995$, the SA found the best solution for 17 circulant graphs (ties were not taken

Table 1: 2-page crossings for circulant graphs.

No.	Graphs	GA_2P	GA_2Ptg	RND	BB+len	SA,RND $\alpha = 0.9$	SA,BB+len $\alpha = 0.9$	SA,BB+len $\alpha = 0.995$	SA,BB+len $\alpha = 0.999$
1	$C_{20}(1,2)$	2	0	128	0	0	0	0	0
2	$C_{20}(1,2,3)$	18	18	244	36	18	18	18	18
3	$C_{20}(1,2,3,4)$	68	68	485	98	83	70	66	64
4	$C_{22}(1,2)$	0	0	136	0	2	0	0	0
5	$C_{22}(1,2,3)$	22	20	336	40	24	20	18	18
6	$C_{22}(1,3,5,7)$	166	172	494	268	253	197	190	179
7	$C_{24}(1,3)$	9	9	185	22	9	10	8	8
8	$C_{24}(1,3,5)$	60	62	387	102	64	67	54	58
9	$C_{24}(1,3,5,7)$	219	193	869	292	221	217	202	197
10	$C_{26}(1,3)$	10	10	199	24	10	11	10	10
11	$C_{26}(1,3,5)$	63	64	426	112	70	73	62	62
12	$C_{26}(1,4,7,9)$	316	290	811	442	385	319	303	299
13	$C_{28}(1,3)$	12	11	221	26	10	11	10	10
14	$C_{28}(1,3,5)$	79	75	632	120	91	79	74	72
15	$C_{28}(1,2,3,4)$	97	103	968	138	112	105	90	88
16	$C_{28}(1,3,5,7,9)$	524	508	1375	828	648	551	551	515
17	$C_{30}(1,3,5)$	84	83	768	130	82	85	74	75
18	$C_{30}(1,3,5,8)$	252	226	1032	400	271	263	254	236
19	$C_{30}(1,2,4,5,7)$	352	346	1716	512	362	381	334	340
20	$C_{32}(1,2,4,6)$	124	134	1182	160	180	150	142	140
21	$C_{34}(1,3,5)$	96	96	783	148	88	99	82	83
22	$C_{34}(1,4,8,12)$	286	302	1420	722	396	550	469	447
23	$C_{36}(1,2,4)$	36	42	927	36	36	36	36	36
24	$C_{36}(1,3,5,7)$	317	301	1462	444	316	311	313	303
25	$C_{38}(1,7)$	36	37	351	108	43	49	38	38
26	$C_{38}(1,4,7)$	149	155	1038	260	158	175	144	137
27	$C_{40}(1,5)$	32	36	545	76	35	34	26	25
28	$C_{42}(1,4)$	24	28	514	42	29	25	23	23
29	$C_{42}(1,3,6)$	109	106	1168	208	149	122	106	97
30	$C_{42}(1,2,4,6)$	168	174	2206	210	197	188	190	184
31	$C_{44}(1,4,5)$	99	112	1255	208	139	110	90	88
32	$C_{44}(1,4,7,10)$	505	491	2309	866	722	636	533	520
33	$C_{46}(1,4)$	31	29	621	46	29	29	26	24
34	$C_{46}(1,5,8)$	268	246	1531	345	250	243	225	225

Table 2: 2-page crossings for Rome graphs.

No.	Graphs	GA_2P	GA_2Ptg	RND	BB+len	SA,RND $\alpha = 0.9$	SA,BB+len $\alpha = 0.9$	SA,BB+len $\alpha = 0.995$	SA,BB+len $\alpha = 0.999$
1	ug1.50	14	14	378	14	4	9	0	0
2	ug2.50	7	6	352	32	4	9	1	0
3	ug3.50	10	7	396	26	1	4	0	0
4	ug4.50	22	13	352	26	4	6	2	0
5	ug5.50	11	12	241	14	5	3	0	0
6	ug6.50	6	6	399	26	6	3	2	0
7	ug7.50	18	15	425	27	15	12	2	0
8	ug8.50	4	8	333	20	1	4	0	0
9	ug9.50	12	8	387	33	6	6	0	0
10	ug10.50	18	16	408	54	16	15	1	1
11	ug11.50	7	8	215	8	6	6	0	0
12	ug12.50	6	5	340	47	3	10	0	0
13	ug13.50	5	12	299	33	7	5	0	0
14	ug14.50	8	14	414	21	7	10	0	0
15	ug15.50	7	7	322	55	5	7	0	0
16	ug16.50	6	23	515	52	12	7	5	0
17	ug17.50	15	12	297	17	0	2	1	0
18	ug18.50	10	14	397	39	3	11	0	0
19	ug19.50	7	23	342	43	10	6	0	0
20	ug20.50	14	3	315	25	4	8	2	0

into account), when the GA algorithms found respectively 5 and 6 of the best solutions. The best solution, with 8 crossings, of the SA algorithm for graph $C_{24}(1,3)$ is depicted in Figure 2.

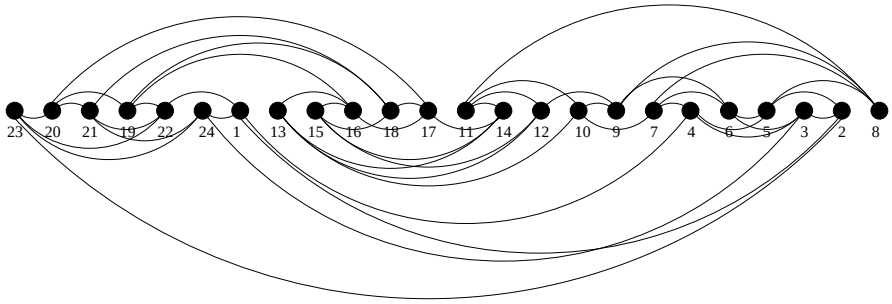


Figure 2: The best solution of the SA algorithm for $C_{24}(1,3)$ with 8 crossings.

For Rome graphs the simulated annealing found clearly better approximations. When α was set to 0.995, all the SA solutions were better than the GA solutions.

Running times of the current implementation of the SA depends highly on the chosen parameters. With $\alpha = 0.99$ the SA runs in 0.14 seconds for graph $C_{20}(1,2)$ and in 484.50 seconds for graph $C_{44}(1,4,7,10)$. When $\alpha = 0.999$, it takes more than 50000 seconds to approximate $C_{44}(1,4,7,10)$. Usually the best solution is found before reaching the frozen temperature, but we did not conduct tests for this.

5. Conclusions and future work

Our experiments showed that simulated annealing is a possible choice when considering approximation methods for the 2-page drawing problem.

The implementation of the algorithm was not time optimal. The tested graph sets were quite small and contained only circulant graphs and Rome graphs. Hence, more comprehensive experiments should be done in the future.

In the inner while loop of Algorithm 1 we first tried to move edges between different sets (r times) and then move vertices on the spine to better positions (r times). Probably there are phases when only edge moves should be considered and phases when only vertex moves should be considered. We have not found any good method to recognise the situations when different kinds of moves should be used. However, it might give better solutions if we select edges (vertices) to be moved using some clever criterion, perhaps edges (vertices) that cause a large number of crossings should have a higher probability to be selected and moved.

The presented optimisation scheme is applicable with small modifications for the book thickness and the k-page crossing number problem.

References

[1] Baur, M. and Brandes, U., Crossing reduction in circular layouts, in: *Proc. 30th Intl. Workshop on Graph-Theoretic Concepts in Computer-Science* (WG '04), LNCS **3353**, Springer, (2004), pp. 332-343.

[2] Buchheim, C. and Zheng, L., Fixed linear crossing minimization by reduction to the maximum cut problem, in *Proc 12th Ann. Int. Computing and Combinatorics Conference* (COCOON'06), to appear.

[3] Cimikowski, R., Algorithms for the fixed linear crossing number problem, *Discrete Applied Mathematics* **122** (2002), pp. 93-115.

[4] Cimikowski, R. and Shope, P., A neural network algorithm for a graph layout problem. *IEEE Trans. on Neural Networks* **7**(2) (1996), pp. 341-346.

[5] GDToolkit: `http://www.dia.uniroma3.it/~gdt/`.

[6] He, H. and Sýkora, O, New circular drawing algorihtms, in: *Proc. Workshop on Information Technologies - Applications and Theory* (ITAT'04), (2004).

[7] He, H., Sýkora, O. and Mäkinen, E., Genetic algorithms for the 2-page book crossing number problem of graphs, *Journal of Heuristics*, (2006), to appear.

[8] He, H., Sýkora, O. and Mäkinen, E., An improved neural network model for the 2-page crossing number problem, *IEEE Trans. on Neural Networks*, (2006), to appear.

[9] He, H., Sýkora, O. and Vrt'o, I., Heuristic crossing minimisation algorithms for 2-page drawings (extended abstract), *The Electronic Notes in Discrete Mathematics* **22** (2005), pp. 527–534.

[10] Kirkpatrick, S., Gelatt, C. and Vecchi, M., Optimization by simulated annealing, *Science* **220** (1983), pp. 671-80.

[11] Masuda, S., Kashiwabara, T., Nakajima, K. and Fujisawa, T., On the NP-completeness of a computer network layout problem, in: Proc. *IEEE Intl. Symposium on Circuits and Systems 1987*, IEEE Computer Society Press, Los Alamitos, (1987), pp. 292-295.

[12] Masuda, S., Kashiwabara, T., Nakajima, K. and Fujisawa, T., Crossing minimization in linear embeddings of graphs, *IEEE Trans. Comput.* **39** (1990), pp. 124-127.

[13] Melikov, A.N., Koreičik V.M. and Tiščenko, V.A., Minimization of the number of intersections of edges of a graph (Russian), *Vyčhislitel'nye Sistemy*, Vyp. **47** (1971), pp. 32-40.

[14] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E., Equation of state calculation by fast computing machines, *Journal of Chemical Physics*, **21** (1953), pp. 1087–1091.

[15] Mäkinen, E., On circular layouts, *International Journal of Computer Mathematics* **24** (1988), pp. 29-37.

[16] Nicholson, T.A.J., Permutation procedure for minimazing the number of crossings in a network, in: *Proc. IEEE* **115**(1) (1968), pp. 21-26.

[17] Shahrokhi, F., Sýkora, O., Székely, L.A. and Vrt'o, I., The book crossing number of a graph, *Journal of Graph Theory* **21** (1996), pp. 413–424.

[18] Six, J.M. and Tollis, I.G., A framework and algorithms for circular drawings of graphs, *Journal of Discrete Algorithms* **4** (2006), pp. 57-73.

[19] Winterbach, W., The crossing number of a graph in the plane, *Master's Thesis*, Dept. Appl. Math., University of Stellenbosch, SA, 2005, pp. 130,173.